

S/N To be assigned

PATENT

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

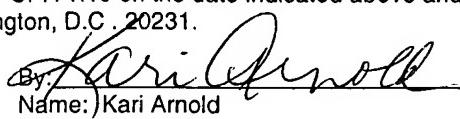
Applicant: Oksanen Serial No.: To be assigned  
Filed: CONCURRENT HEREWITH Docket No.: 796.417USW1  
Title: METHOD FOR RECOVERING A DATABASE PROVIDED WITH DISK BACK-UP

CERTIFICATE UNDER 37 CFR 1.10

'Express Mail' mailing label number: EL887039171US

Date of Deposit: December 10, 2001

I hereby certify that this correspondence is being deposited with the United States Postal Service 'Express Mail Post Office To Addressee' service under 37 CFR 1.10 on the date indicated above and is addressed to the Assistant Commissioner for Patents, Washington, D.C. 20231.

  
By: \_\_\_\_\_  
Name: Kari Arnold

JC973 U.S. PTO  
10/015131  
12/10/01

**SUBMISSION OF PRIORITY DOCUMENT**

Box Patent Application  
Assistant Commissioner for Patents  
Washington, D.C. 20231

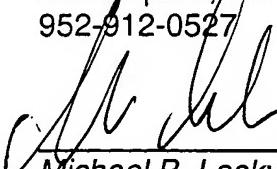
Dear Sir:

Enclosed is a certified copy of Finnish application, Serial Number 991336, filed 10 June 1999, the priority of which is claimed under 35 U.S.C. §119.

Respectfully submitted,

Altera Law Group, LLC  
6500 City West Parkway – Suite 100  
Minneapolis, MN 55344-7701  
952-912-0527

Date: December 10, 2001

By: 

Michael B. Lasky  
Reg. No. 29,555  
MBL/blj

Helsinki 23.10.2001

U.S. PRO  
10/015131  
12/10/01  
sc973

E T U O I K E U S T O D I S T U S  
P R I O R I T Y D O C U M E N T



Hakija  
Applicant

Nokia Telecommunications Oy  
Helsinki

Patentihakemus nro  
Patent application no

991336

Tekemispäivä  
Filing date

10.06.1999

Kansainvälinen luokka  
International class

G06F 11/14

Keksinnön nimitys  
Title of invention

"Menetelmä levyvarmennetun tietokannan elvyttämiseksi"

Hakijan nimi on hakemusdiaariin 30.01.2000 tehdyin nimenmuutoksen jälkeen **Nokia Networks Oy**.

The application has according to an entry made in the register of patent applications on 30.01.2000 with the name changed into **Nokia Networks Oy**.

Täten todistetaan, että oheiset asiakirjat ovat tarkkoja jäljennöksiä patentti- ja rekisterihallitukselle alkuaan annetuista selityksestä, patenttivaatimuksista, tiivistelmästä ja piirustuksista.

This is to certify that the annexed documents are true copies of the description, claims, abstract and drawings originally filed with the Finnish Patent Office.

Pirjo Kaila  
Tutkimussihteeri

Maksu 300,- mk  
Fee 300,- FIM

*Maksu perustuu kauppa- ja teollisuusministeriön antamaan asetukseen 1782/1995 Patentti- ja rekisterihallituksen maksullisista suoritteista muutoksineen.*

*The fee is based on the Decree with amendments of the Ministry of Trade and Industry No. 1782/1995 concerning the chargeable services of the National Board of Patents and Registration of Finland.*

## **Menetelmä levyvarmennetun tietokannan elvyttämiseksi**

### **Keksinnön ala**

Keksintö liittyy yleisesti levyvarmennettuun muistiin, jossa suoritetaan roskankeroita. Tarkemmin sanottuna, eksintö koskee menetelmää, jonka avulla funktioalinen keskusmuistitietokanta, jossa suoritetaan roskankeroita, saadaan toipumaan vikatilanteen, esim. sähkökatkon jälkeen. Funktioalaisuudella tarkoitetaan tässä yhteydessä sitä, että muistia käyttävä sovellus ei saa tehdä muutoksia suoraan tietoon, vaan talletetusta tiedosta on ensin otettava kopio, johon muutos tehdään (muutosta ei tehdä suoraan olemassa olevaan tietoon). Tällaista päivitysmenettelyä kutsutaan myös nimellä "copy-on-write". Keksinnön mukaista menetelmää voidaan käyttää useissa erilaisissa tietokantaympäristöissä, erityisesti sellaisissa, joissa varmennus ja reaalialkaisuus ovat tärkeitä tekijöitä. Esimerkki tällaisesta on puhe-linkeskuksessa oleva tietokanta, joka sisältää tietoliikenneverkon toimintaan liittyvää dataa.

### **Keksinnön tausta**

Sovellusohjelman käyttöön dynaamisesti alllokoidut muistialkiot muuttuvat yleensä, sovellusohjelman toiminnan johdosta, ajan myötä sellaisiksi, että ne eivät ole enää sovellusohjelman saavutettavissa. Muistialkio katsoaan roskaksi (garbage), jos sovellusohjelma ei enää saavuta sitä minkäänlaista reittiä pitkin. Roskankeroulla (garbage collection) tarkoitetaan sellaista automaattista järjestelmää, jonka avulla vapautetaan tällaisten muistialkioiden viemä muistitalo uudelleen sovellusohjelman käyttöön. Roskancerääjän on siis erotettava roskat elossa olevien (live) muistialkioiden joukosta. Viimemainituilla tarkoitetaan sellaisia muistialkioita, jotka ovat vielä sovellusohjelman saavutettavissa ja joita sovellusohjelma tulee mahdollisesti vielä käyttämään. Roskancerääjät määrittelevät elossa olevat muistialkiot tyypillisesti siten, että ne ovat niitä muistialkioita, jotka ovat saavutettavissa muistin juurijoukosta (root set) seuraamalla muistiin talletettuja osoittimia. Juurijoukolla tarkoitetaan sellaista dataa, joka on välittömästi ohjelman käytettävissä, minkäänlaista osoitinketjua seuraamatta.

Roskankeroon lisää ohjelmien luotettavuutta, sillä sen avulla pystytään muun muassa eliminoimaan muistinhallintavirheitä.

Kopioivassa roskankерuussа ei itse asiassa kerätä roskia, vaan elossa olevat muistialkiot siirretään (kopioidaan) niiden sijaintialueelta uudelle muistialueelle, joka on yleensä yhtenäinen (contiguous). Alkuperäinen muistialue voidaan tämän jälkeen vapauttaa sovellusohjelman käyttöön, koska kaikki 5 ko. alueella elossa olleet muistialkiot on siirretty muualle. Kopioivia roskankерuumenetelmiä pidetään yleisesti ottaen varsin tehokkaina. Niiden eräs tärkeä etu on se, että ne ovat kompaktovia, jolloin muistin sirpaloituminen (fragmenteerituminen) voidaan estää.

Nykyisin yleinen suuntaus on käyttää sukupolvittain tapahtuvaa 10 roskankерuuta. Tällaisessa järjestelmässä jaetaan muisti useisiin alueisiin iän mukaan (sukupolvet) ja suoritetaan keruu nuoremmissa sukupolvissa useammin kuin vanhemmissa sukupolvissa, koska nuoremmissa sukupolvissa muistialkiot muuttuvat roskaksi todennäköisemmin (nopeammin) kuin vanhoissa 15 sukupolvissa. Näin ollen nuorempien sukupolvien keruun avulla saadaan muistia vapautettua tehokkaammin (samaa työmääräää kohti enemmän vapautettua muistia). Ei-sukupolvittaisten kopioivien roskankерuumenetelmien periaatteellisena epäkohtana on myös se, että samoja muistialkioita joudutaan 20 kopioimaan uudelleen jokaisella keräyskerralla, koska osa muistialkioista säilyy elossa keräyskerrasta toiseen. Tällöin osa roskankерääjän ajasta kuluu siihen, että se kopioi toistuvasti samoja muistialkioita. Sukupolvittain tapahtuvalla roskankерuulla voidaan välttää myös suuri osa tällaisesta toistuvasta kopioinnista.

Esillä oleva keksintö liittyy tällaiseen sukupolvittain tapahtuvaan 25 roskankерuuseen keskusmuistitietokannassa, joka on lisäksi toipuva (recoverable). Keskusmuistitietokannalla, tai lyhyesti tietokannalla, tarkoitetaan tässä yhteydessä tiettyä keskusmuistialuetta, joka on yleensä yhtenäinen. Toipuminen tarkoittaa puolestaan sitä, että keskusmuistissa olevaa dataa kirjoitetaan myös levymuistiin, jolloin vikatilanteen, esim. sähkökatkon 30 seurauksena voidaan rekonstruoida vikatilanteen hetkellä keskusmuistissa vallinnut tilanne levymuistiin talletetun datan avulla.

Keksinnön mukainen menetelmä on tarkoitettu erityisesti sellaisille sovelluksille, joissa reaalialaisuus ja varmuus ovat tärkeitä ominaisuuksia. Kuten edellä mainittiin, erään sovellusalueen muodostavat tietoliikenneverkon 35 verkkoelementit. Esim. puhelinkeskuksissa on pystytävä suorittamaan roskankeroon mahdollisimman tehokkaasti niin, että se ei aiheuta ylimääräistä

viiettä olemassa oleville yhteyksille tai käynnissä oleville yhteydenmuodos-tusprosesseille.

Tunnetuissa levyvarmennetuissa järjestelmissä ylläpidetään levy-varmennettuina ns. muistojoukkoja (remembered sets) ja mahdollisesti vielä lukisia muitakin apurakenteita mahdollisen vikatilanteen varalta, jotta tietokanta voitaisiin rekonstruoida vikatilanteen jälkeen. Tällaisissa järjestelmissä on roskankerousta aiheutuva levykirjoituksen määrä erittäin suuri. Toipumisen yhteydessä esim. lokeihin talletetut muistojoukot ja muut apurakenteet käydään läpi ja päivitetään muun tietokannan tila vikatilannetta edeltävää vastavaksi. Tällaisten tunnettujen järjestelmien epäkohtana on myös se, varmen-nuksesta aiheutuva datamäärä on suuri verrattuna hyötydatan käytössä olevaan muistikapasiteettiin.

Mahdollisimman suurta reaalialkaisuutta vaativissa levyvarmennetuissa järjestelmissä eräs huomattava ongelma on siis se, kuinka varmennuksesta johtuva levyliikenne (lähinnä levylle tapahtuva kirjoitus) tulisi hoitaa niin, että siitä aiheutuu mahdollisimman vähän viivettä järjestelmän muuhun toimintaan. Varmennuksen kannalta olisi luonnollisestikin suotavaa, jos keskusmuistissa tapahtuvat muutokset voitaisiin viedä heti levylle, mutta järjestelmän tehokkuuden kannalta tämä ei ole mahdollista, sillä käytännössä levyliikenne kasvaisi aivan liian suureksi ja hidastaisi liiaksi järjestelmän muuta toimintaa. Järjestelmän tehokkuuden kannalta ovat erityisesti usein esiintyvät pienet muutokset ongelmallisia, sillä levymuiston toiminta hidastuu huomattavasti, jos kirjoituspäätä joudutaan toistuvasti siirtelemään paikasta toiseen. Levymuistiin olisikin pyrittävä kirjoittamaan kerralla mahdollisimman paljon peräkkäistä dataa. Levyliikenteen optimoimiseksi on siis kiinnitettävä huomiota paitasi levyliikenteen määrään, myös siihen, että levylle tapahtuvat kirjoitukset pystytään suorittamaan mahdollisimman yhtenäisinä sekä pitämään kirjoituskerrat vähäisinä.

Ongelmana on siis se, miten roskankerouta tulisi suorittaa, jotta suuri keskusmuistialue saataisiin levyvarmennettua niin, että järjestelmän suorituskyky pysyy mahdollisimman hyvänä.

### **Keksinnön yhteenveto**

Keksinnön tarkoituksesta on saada aikaan ratkaisu edellä kuvattuun ongelmaan. Tämä päämäärä saavutetaan menetelmällä, joka on määritelty itsenäisessä patenttivaatimuksessa.

Keksinnön mukaisessa menetelmässä tietokantaan liitettävät uudet solut sijoitetaan aluksi ns. ensimmäiseen sukupolveen, joka on muuhun tietokantaan nähdyn erillinen muistin osa. Ajoittain, tai kun ensimmäiselle sukupolvelle varattu muistitila ehtyy, ensimmäinen sukupolvi kerätään nuorimmaksi 5 sukupolveksi ns. kypsien sukupolvien (mature generation) alueelle. Ajatuksena on pitää levyllä kunkin (kypsän) sukupolviversion lisäksi kyseisen sukupolven edellistä versiota ja käyttää sitä hyväksi toipumisessa niin, että viimeisin versio saadaan päivitettyä oikeaksi. Toipuminen toteutetaan mainittujen versioiden 10 avulla (a) rekonstruoimalla mainitun edellisen levylle talletetun version muisto-joukko, joka osoittaa ne osoittimet, joista on viittauksia kyseiseen sukupolveen ja (b) muuttamalla muistojoukon osoittamat osoittimet osoittamaan mainitun viimeisimmän version muistisoluihin. Levylle ei tarvitse lainkaan kirjoittaa sellaisia muutoksia, jotka roskankeruun edetessä muuttavat ko. sukupolvesta (viimeisimmästä versiosta) muihin sukupolviin olevia viittauksia. Keksinnössä 15 käytetään siis hyväksi sellaista dataa (edellistä sukupolviversiota), joka joutuu taan joka tapauksessa kirjoittamaan levylle, niin, että viimeisin versio voidaan jättää vanhentuneeksi ja tällä tavoin vähentää levyllä tapahtuvaa kirjoitusta.

Keksinnön mukainen ratkaisu mahdollistaa siis varmennuksen takia suoritettavien levykirjoitusten oleellisen vähentämisen, jolloin tietokannan 20 toiminta saadaan oleellisesti aikaisempaa nopeammaksi.

Keksinnön mukaisen ratkaisun ansiosta saadaan myös varsinaiselle hyötydatalle jäävä muistitila entistä oleellisesti suuremmaksi, koska muistojoukkoja ei tarvitse lainkaan tallettaa levylle. Keskusmuistissakin tarvitaan muistojoukkoja vain keräämättömässä kypsissä sukupolvissa. 25

### **Kuvioluettelo**

Seuraavassa keksintöä ja sen edullisia toteutustapoja kuvataan tarkemmin viitaten kuvioiden 1-8c mukaisiin esimerkkeihin oheissa piirustuksissa, joissa 30

kuvio 1 on lohkokaavio, joka havainnollistaa keksinnön mukaisen järjestelmän osia,

kuvio 2 havainnollistaa keskusmuistin kokoonpanoa,  
kuvio 3 on vuokaavio, joka havainnollistaa keksinnön mukaisen järjestelmän 35 toimintaa,

kuviot 4a ja 4b havainnollistavat kuviossa 3 esitetyjä menetelmävaiheita,

- kuviot 5a...5f havainnollistavat ensimmäisen sukupolven keruun eri vaiheita,  
 kuvioissa 6a...6e havainnollistavat kypsien sukupolvien keruun eri vaiheita,  
 kuva 7 on esimerkki keskusmuistin ja levymuistin välisestä eroasta vikatilan-  
 teessa, ja
- 5 kuviot 8a...8c havainnollistavat keksinnön mukaisia vaiheita oikean tietokanta-  
 kuvan rekonstruoimiseksi.

### **Keksinnön yksityiskohtainen kuvaus**

Kuviossa 1 on esitetty keksinnön mukaisen menetelmän toteutusympäristöä. Esim. puhelinkeskukseissa EX on levyvarmennettu keskusmuistitietokanta CB, jota hallitaan keksinnön mukaisella muistinhallintajärjestelmällä MMS. Muistinhallintajärjestelmä kommunikoi käyttöjärjestelmän OS välityksellä laitteiston olion (levymuistin DM ja keskusmuistin RAM) kanssa. Sovelluksiksi tai sovellusohjelmiksi (APP) kutsutaan tässä yhteydessä kaikkia niitä kerroksia, jotka ovat muistinhallintajärjestelmän yläpuolella, sillä ne voidaan katsoa muistinhallintajärjestelmän kannalta sovelluksiksi. Näihin kerroksiin voi varsinaisen sovelluksen lisäksi kuulua esim. hakurakenteita (index structures). Esillä oleva keksintö ei koske näitä kerroksia, eikä sen toteutus riipu näiden kerrosten toteutuksesta.

10 Kuten edellä mainittiin, (keskusmuisti)tietokannalla tarkoitetaan tässä yhteydessä yhtenäistä keskusmuistialuetta. Kuvossa 2 on havainnollistettu keskusmuistin kokoonpanoa. Tietokanta koostuu juurilohkosta, ensimmäisestä sukupolvesta (1 kpl) sekä kypsistä sukupolvista, joita on vaihteleva määrä tilanteesta riippuen. Suuressa tietokannassa voi olla tuhansia kypsiä sukupolia, kun taas tyhjässä tietokannassa ei välttämättä ole vielä yhtään kypsää sukupolvea. Jokainen kypsä sukupolvi muodostuu yhdestä tai useammasta sivusta P. Vaikka kuviossa on esitetty yhden sukupolven sivut peräkkäin, ovat yhden sukupolven sivut käytännössä peräkkäin vain alussa, kun tietokanta on nuori ja hajaantuneina kypsien sukupolvien muistialueella sen jälkeen, kun roskankeruuta on suoritettu tarpeeksi kauan. Kirjoituspyynnöt ovat sivukohtisia, joten käytännössä levylle kirjoitetaan yksi tai useampi sivu kerrallaan. Sivut ja ensimmäinen sukupolvi sisältävät soluja C, joka on pienin mahdollinen muistin varausyksikkö (kuviossa on selvyyden vuoksi esitetty vain yksi solu). Tietokanta koostuu siis iän mukaan järjestetyistä, vaihtelevan kokoisista sukupolvista. Sukupolvien koko vaihtelee tyypillisesti muutamasta sadasta kilotavusta useisiin megatavuihin, sivut ovat kooltaan tyypillisesti noin 64 kilotavua

- ja solut puolestaan muutamasta sanasta muutamiin kymmeniin sanoihin. Juurilohkossa on tyypillisesti jokin vakiomäärä sanoja, esim. 128 sanaa. Sovellusohjelma muodostaa kaikki muut tietorakenteet solusta. Solujen sisältämä data riippuu sovellusohjelmasta ja voi olla minkä tahansa typpistä dataa.
- 5 Roskankerääjän on kuitenkin oltava tietoinen siitä, mitkä sanat solussa ovat osoittimia ja mitkä eivät ja lisäksi siitä, minkä kokoinen solu on.

Kuva 3 on vuokaavio, joka havainnollistaa kuvion 1 mukaisen järjestelmän toimintaa. Sovellusohjelma suorittaa omia tehtäviään ja kun se tarvitsee lisää muistia, se pyytää muistinhallintajärjestelmää allokoimaan 10 muistia käyttöönsä (vaihe 31). Muistinhallintajärjestelmä antaa sovelluksen käyttöön muistia ensimmäisen sukupolven alueelta. Tämä toiminta jatkuu niin kauan kunnes ensimmäinen sukupolvi on täynnä tai sen täyttöaste ylittää tietyn raja-arvon. Tämän jälkeen muistinhallintajärjestelmä kerää ensimmäisen sukupolven (vaihe 33). Keräys voi lähteä käyntiin myös tietyn väliajoin, esim. 15 kun vaiheessa 32 on havaittu, että ajastin on lauennut. Ensimmäisen sukupolven keräys käsittää seuraavat toimenpiteet. Aluksi muistinhallintajärjestelmä luo uuden kypsän sukupolven, minkä jälkeen se kopio tähän uuteen kypsään sukupolveen ensimmäisestä sukupolvesta ne solut, jotka ovat vielä elossa. Ne 20 osoittimet, jotka olivat juurilohkosta kerättyihin soluihin on päivitetvä osoittamaan solujen uuteen sijaintipaikkaan uudessa kypsässä sukupolvessa. Lopuksi uusi kypsä sukupolvi on kirjoitettava levylle. Kuvioissa 4a ja 4b on havainnollistettu keräysvaihetta esittämällä neljä solua C1...C4. Keräyksen alkaessa solut ovat ensimmäisessä sukupolvessa (kuvio 4a). Soluista kopioidaan uuteen kypsään sukupolveen ne, jotka eivät ole roskia (C1, C3 ja C4).

25 Juurilohkosta ensimmäisen sukupolven soluihin olevat osoittimet päivitetään osoittamaan ko. solujen uusiin sijaintikohtiin. Kerätty data (uusi kypsä sukupolvi) varmennetaan kirjoittamalla se levylle (nuoli A1). Kun nämä uudet sivut, jotka otettiin käyttöön kypsien sukupolvien alueesta on kirjoitettu levylle, kirjotetaan myös juurilohko levylle (vaihe 34b ja nuoli A2) määrättyn sijaintikohdan. Juurilohko on pakko kirjoittaa levylle vasta ko. sivujen jälkeen, koska pääinvastaisessa tapauksessa saattaisi käydä niin, että vian esiintyessä heti juurilohkon kirjoituksen jälkeen levyllä oleva juurilohko viittaisi sellaiseen dataan, jota ei ole olemassa. Lisäksi juurilohko on edullista kirjoittaa levylle kahteen kertaan (duplicated writing), koska tällainen varmennus on käytännössä 30 hyvin toteutettavissa juurilohkon pienuuden vuoksi. Lopuksi tyhjennetään 35

ensimmäinen sukupolvi, jolloin siitä voidaan taas allokoida muistia sovelluksen käyttöön.

Jos tämän jälkeen syntyy vikatilanne, esim. sähkökatkos, voidaan äsknen kirjoitettu data ja kypsissä sukupolvissa oleva vanhempi data kirjoittaa 5 takaisin levyltä.

Vähitellen myös kypsien sukupolvien alue täytyy ja osa siellä ole- 10 vista solusta muuttuu roskiksi. Tämän takia testataan vaiheessa 35, tarvitaan-ko roskankeroon kypsien sukupolvien alueelta. Mikäli roskia ei tarvitse kerätä, palautetaan ohjaus sovellukselle (palataan vaiheeseen 31). Muussa tapauk- 15 sessa suoritetaan roskankeroon kypsien sukupolvien alueelta (vaihe 36). Kun tästä keruuta on suoritettu, palautetaan ohjaus taas sovellukselle. Käytännössä roskia on kerättävä kypsien sukupolvien alueelta vähitellen pienissä erissä, koska kypsien sukupolvien alue on niin iso, että koko alueen keruu kerralla veisi liikaa aikaa ja aiheuttaisi viivettä järjestelmän muuhun toimintaan. Myös 20 kypsat sukopvet kirjoitetaan levylle sen jälkeen, kun niistä on kerätty roskat. Tämä levykirjoitus voi tapahtua kokonaan kuvion vaiheessa 36 tai se voi jatkua vielä sen jälkeen, kun ohjaus on palautettu sovellukselle, kuitenkin päättyn ennen seuraavan juurilohkon kirjoitusta. Tämän takia järjestelmä voi odottaa 25 ennen juurilohkon kirjoitusta, että kaikki aikaisemmat levykirjoitukset ovat valmiit (vaihe 34a). Kypsat sukopvet kerätään nuoremista vanhempiin sukupolviin päin.

Kuten vuokaaviosta havaitaan, keksinnön mukaisessa menetel- 30 mässä roskankeroon tapahtuu ns. stop&copy-periaatteen mukaisesti eli siten, että lisämuistilan saamiseksi sovellus pysähtyy ja kopioiva roskankeroon käynnistyy. Perinteisestä stop&copy-menetelmistä poiketen esillä olevassa keksinnössä ei kuitenkaan tarvita kaksinkertaista muistitilaan muiden kuin kulloinkin keruun kohteena olevan sukupolven osalta. Juurilohkossa ja kypsien sukupolvien alueella oleva data on levyvarmenettu, kulloinkin ensimmäisessä sukupolvessa olevaa dataa ei sen sijaan ole varmenettu (käytännössä en- 35 simmäisen sukupolven keruun tapahtuu esim. 2...10 kertaa sekunnissa).

Kuvioissa 5a...5f on havainnollistettu tarkemmin ensimmäisen su- 35 kupolven keruun eri vaiheita. Selvyden vuoksi kuvioissa on esitetty vain soluja ja sivut on jätetty esittämättä. Kuten edellä mainittiin, kypsien sukupolvien alue koostuu todellisuudessa sivuista, joita otetaan käyttöön tarpeen mu- kaan. Solut on esitetty suorakulmioina, jotka on numeroitu yhdestä neljään. Juurilohkosta on tässä tapauksessa osoittimet soluun 1 ja soluun 4. Lisäksi

kuvioissa on esitetty aputietorakenteena toimiva kopiopino CS, johon on talletettu sellaisten osoittimien osoitteet, jotka osoittavat soluihin, joita ei ole vielä kopioitu.

- Kuviossa 5a on esitetty oletettu alkutilanne, jossa kaikki solut ovat ensimmäisessä sukupolvessa ja kopiopino on tyhjä. Keruu alkaa siten, että juurilohko käydään ensin läpi ja kopioidaan kypsien sukupolvien alueelle ne solut, joihin juurilohkosta (ei esitetty) osoitetaan suoraan. Tässä tapauksessa kopioidaan siis solut 1 ja 4. Samalla kirjoitetaan näiden solujen vanhaan kohtaan osoitin osoittamaan, mihin kohtaan solu siirrettiin. Näistä osoittimista käytetään englanninkielistä termiä "forward pointer" ja ne ovat siis osoittimia solun vanhasta sijaintipaikasta solun uuteen sijaintipaikkaan. Forward-osoitimia on kuvioissa merkitty katkoviivoilla. Lisäksi juuriosoitimet (vinot nuolet) muutetaan osoittamaan solujen uuteen sijaintiin. Samalla kun solut kopioidaan, tutkitaan niiden sisältämät osoittimet ja (kopioiduissa) soluissa olevien osoittimien osoitteet kirjoitetaan kopiopinoon. Tässä tapauksessa havaitaan ensinnäkin, että solussa 1 on osoitin soluun 2. Tällöin kopiopinoon talletetaan osoitin siihen kohtaan solua 1, jossa on osoitin soluun 2 (pinon alimmainen alkio kuviossa 5b). Vastaavasti havaitaan, että solussa 4 on osoitin soluun 3, jolloin kopiopinoon talletetaan osoitin siihen kohtaan solua 4, josta on osoitin soluun 3 (pinon ylimmäinen alkio kuviossa 5b). Näin ollaan kuvion 5b mukaisessa tilanteessa.

- Tämän jälkeen käydään kopiopinoa läpi ylhäältä alkaen kunnes pino on tyhjä. Aluksi pinosta luetaan siis osoitin solun 4 tiettyyn kohtaan. Tästä kohdasta löydetään osoitin soluun 3, joka kopioidaan ensimmäisestä sukupolvesta kypsiin sukupolviin. Solun 4 sisältämä osoitin asetetaan osoittamaan solun 3 uuteen sijaintikohtaan ja solun 3 vanhaan sijaintikohtaan kirjoitetaan osoitin osoittamaan uuteen sijaintikohtaan (forward pointer). Nämä on päästy kuvion 5c esittämään tilanteeseen. Tämän jälkeen luetaan kopiopinosta osoitin solun 1 tiettyyn kohtaan. Tästä kohdasta löydetään osoitin soluun 2, joka kopioidaan ensimmäisestä sukupolvesta kypsiin sukupolviin. Solun 1 sisältämä osoitin asetetaan osoittamaan solun 2 uuteen sijaintikohtaan ja solun 2 vanhaan sijaintikohtaan kirjoitetaan osoitin osoittamaan uuteen sijaintikohtaan (forward pointer). Samalla kun solu 2 kopioitiin, havaittiin, että solu sisältää osoittimen soluun 3. Tällöin kopiopinoon lisätään osoitin siihen kohtaan solua 2, josta on osoitin soluun 3. Kopiopino ehti siis olla hetken aikaa tyhjä, mutta

koska kopioitavassa solussa 2 oli osoitin, kirjoitettiin ko. osoittimen osoite kopiopinoon. Näin on päästy kuvion 5d esittämään tilanteeseen.

Tämän jälkeen luetaan taas kopiopino tyhjäksi. Näin edetään solun 2 kautta soluun 3, jolloin solussa jo olevan forward-osoittimen perusteella huomataan, että solu 3 on jo kopioitu kypsiin sukupolviin. Seuraamalla forward-osoitinta päästään solun 3 uuteen sijaintikohtaan, jolloin voidaan muuttaa solusta 2 oleva osoitin solun 3 uuteen sijaintikohtaan. Näin on päästy kuvion 5e esittämään tilanteeseen, jossa kaikki ensimmäisessä sukupolvessa olleet solut on kopioitu kypsiin sukupolviin ja lisäksi kopiopino on tyhjä. Tämän jälkeen voidaan kirjoittaa kypsiin sukupolvien uudet sivut (solut 1...4) (ja juurilohko) levylle sekä vapauttaa ensimmäinen sukupolvi uudelleen käyttöön. Näin ollaan kuvion 5f lopputilanteessa, jossa elossa olevat solut on siirretty kypsiin sukupolvien alueelle, ne on kirjoitettu levylle (ei esitetty) ja ensimmäinen sukupolvi on vapautettu. Keruuprosessissa tarkastetaan siis ennen kopointia, onko solusta forward-osoitin ja kopointi suoritetaan vain jos ko. osoitinta ei ole.

Kun kypsiin sukupolvien alueelta otetaan käyttöön uusia sivuja, täytyy ko. alue vähitellen ja lisäksi alueella olevat solut muuttuvat vähitellen roskiksi. Tämän vuoksi on jossain vaiheessa suoritettava roskien keruuta myös kypsiin sukupolvien alueelta. Kuvioissa 6a...6e on havainnollistettu roskankereuta kypsiin sukupolvien alueelta. Kuvioissa on kirjaimilla A...G merkityillä suorakulmioilla esitetty eri sukupolia. Kerättävät sukupolvet ovat A...D, joista A on vanhin ja D nuorin sukupolvi. Sukupolvien sisällä olevilla, numeroilla 1...8 merkityillä laatikoilla on puolestaan merkitty soluja. Nuolilla on esitetty osoittimia, kuten aiemminkin.

Kuviossa 6a on esitetty oletettu alkutilanne (keskusmuistissa). Tässä tapauksessa juurilohkosta on osoittimet soluihin 5 ja 6. Kuten kuviosta havaitaan, solu 3 on roskaa. Kun kypsiin sukupolvien roskankeruu alkaa, merkitään ne sukupolvet (A...D), jotka aiotaan kerätä. Oletetaan, että tässä vaiheessa ollaan kuvion 3 vaiheessa 31, jossa muistinhallintajärjestelmä allokoi sovelluksen käyttöön muistia niin kauan, että ensimmäinen sukupolvi tulee täyneen. Kun täyneen tullut ensimmäinen sukupolvi kerätään, päästään kuvion 6b mukaiseen tilanteeseen, jossa kypsiin sukupolvien alueella on lisäksi uusin sukupolvi E, joka sisältää tässä yksinkertaistetussa esimerkkipaiksessa solun 7, johon on osoitin juurilohkosta ja josta on osoitin sukupolven D soluun 6. Ensimmäisen sukupolven keruun yhteydessä havaittiin, että kerätyissä soluissa (solu 7) oli osoitin soluun, joka on sellaisessa sukupolvensa

sa, joka on merkitty kerättäväksi. Kun tällainen osoitin havaitaan, kirjoitetaan viitatuun sukupolven muistojoukkoon (remembered set) ko. osoittimen osoite. Muistojoukkoja on kuvioissa merkitty viitemerkillä RS. Muistojoukko on jokin järjestetty joukko, esim. lista, joka sisältää kahden objektijoukon (sukupolven) 5 väliset viittaukset. Muistojoukkoja käytetään tällä tavoin sukupolvikohtaisesti ilmoittamaan nuoremmista sukupolvista olevat viittaukset kyseiseen sukupol-veen.

Tämän jälkeen kerätään ensiksi kypsä sukupolvi D. Tätä varten al-10 lokoidaan sukupolvi D', johon kerätään sukupolven D elossa olevat muistisolut (kuvioissa on viitemerkin perässä olevalla pilkulla osoitettu sitä, että kysymyk-15 sessä on kopio). Keruu aloitetaan käymällä läpi sukupolven D muistojoukko. Tässä muistojoukossa on nyt viittaus solussa 7 olevaan osoittimeen (vrt. kuvio 6b), jolloin lukemalla ko. osoitin päästään soluun 6. Näin kopioidaan sukupol-20 veen D' alaksi solu 6 (jolloin sukupolvessa D' on alaksi solu 6'). Osoitin solusta 7 vaihdetaan tämän jälkeen osoittamaan soluun 6'. Lisäksi solusta 6 asetetaan forward-osoitin soluun 6' (forward-osoittimia ei ole esitetty kuvioissa). Kun solu 6 kopioitiin soluksi 6', havaittiin lisäksi, että solusta 6' on osoitin soluun (solu 4), joka on sellaisessa sukupolvessa, jota ei ole vielä kerätty, mutta joka on merkitty kerättäväksi. Tällöin lisätään kyseisen sukupolven muistojoukkoon 25 solussa 6' olevan osoittimen osoite.

Kun sukupolven D muistojoukko on käyty läpi, käydään läpi juuri-lohko. Juurilohkosta havaitaan olevan osoitin soluun 5, jolloin solu 5 kopioi-25 daan soluksi 5' sukupolveen D' ja juurilohkosta oleva osoitin muutetaan osoit-tamaan soluun 5' (ja asetetaan forward-osoitin osoittamaan solusta 5 soluun 5'). Kopioitaessa solu 5 havaittiin taas, että solusta 5' on osoitin soluun (solu 4), joka sellaisessa sukupolvessa, jota ei ole vielä kerätty, mutta joka on mer-30 kittä kerättäväksi. Tällöin lisätään kyseisen sukupolven muistojoukkoon solus-sa 5' olevan osoittimen osoite.

Koska kopiopino (ei esitetty) on tässä vaiheessa tyhjä, sukupolves-35 sa D ei ole soluja, joihin viitataan soluista 5 tai 6. Tämän jälkeen kirjoitetaan sukupolvi D' levylle ja vapautetaan edellinen sukupolvi D keskusmuistista, mutta ei vielä levyltä. Näin on päästy kuvion 6c esittämään tilanteeseen, jossa voitaisiin antaa sovelluksen jatkaa työtään. Muistojoukkoja ei talleteta lainkaan levylle, sillä toipumisprosessi osaa päätellä ne itse, kuten jäljempänä kuva-taana.

Oletetaan tässä vaiheessa kuitenkin, että vapaita sivuja tarvitaan edelleen lisää, joten tämän jälkeen kerätään vielä toinen kypsä sukupolvi C sukupolveksi C' ennen kuin taas palautetaan ohjaus sovellusohjelmalle. Ensin käydään taas läpi kerättävän sukupolven muistojoukko. Tällöin päästään soluun 4, joka kopioidaan soluksi 4' sukupolveen C' ja asetetaan forward-osoitin, jolloin solun 4 uudelleenkopiointi estyy, kun soluun 4 edetään toisen kerran sukupolven C muistojoukosta. Osoittimet soluista 6' ja 5' muutetaan osoittamaan soluun 4'. Kun solu 4 kopioitiin, havaittiin, että solussa 4' on osoitin soluun (solu 2), joka sellaisessa sukupolvessa (B), jota ei ole vielä kerätty, mutta joka on merkitty kerättäväksi. Tällöin lisätään kyseisen sukupolven muistojoukkoon solussa 4' olevan osoittimen osoite.

Koska kopointipino on nyt tyhjä (yksittäisen sukupolven keruu jatkuu kunnes kopointipino on tyhjä), tämän jälkeen voidaan kirjoittaa sukupolvi C' levylle ja vapauttaa sukupolvi C keskusmuistista, mutta ei vielä levyltä. Näin on saatu kerättyä kaksi kypsää sukupolvea. Oletetaan, että vapaita sivuja ei tarvita enempää, joten ohjaus voidaan nyt palauttaa sovellusohjelmalle. Kun ensimmäinen sukupolvi tulee täyneen, se kerätään uudeksi kypsäksi sukupolveksi F. Näin on päästy kuvion 6d mukaiseen tilanteeseen.

Jotta kypsiä sukupolia ei muodostuisi liikaa, voidaan niitä myös yhdistää roskankeruun yhteydessä. Yhdistäminen voidaan suorittaa esim. aina silloin, kun kahden sukupolven yhteenlaskettu koko on pienempi kuin tietty ennalta määritetty raja. Kun sukupolvet A ja B kerätään, luodaan uusi kypsä sukupolvi AB'. Ensin käydään taas läpi B:n ja A:n muistojoukot, jolloin huomataan, että solussa 4' on osoitin, joka osoittaa soluun 2. Solu 2 kopioidaan soluksi 2' sukupolveen AB' ja asetetaan forward-osoitin solusta 2 soluun 2'. Osoitin solusta 4' asetetaan osoittamaan soluun 2'. Tässä vaiheessa havaitaan, että kopioitavassa solussa on osoitin soluun, jota ei ole vielä kopioitu, joten kopiopinoon (ei esitetty kuviossa) lisätään osoitin siihen kohtaan solua 2', josta on osoitin soluun 1. Lisäys suoritetaan, koska molempia sukupolia kopioidaan samanaikaisesti (roskankeruun kannalta A ja B ovat sama sukupolvi, jolloin lisäys suoritetaan kopiopinoon).

Kun kopiopino tyhjennetään, saadaan solu 1 kopioitua soluksi 1'. Osoitin solusta 2' asetetaan osoittamaan kopioituun soluun (1').

Tämän jälkeen kirjoitetaan sukupolvi AB' levylle ja vapautetaan sukupolvet A ja B keskusmuistista, mutta ei vielä levyltä. Näin on päästy kuvion 6e mukaiseen tilanteeseen, jossa kaikista kypsistä sukupolvista on kerätty

roskat kertaalleen. Uusissa kypsissä sukupolvissa on sillä välin syntynyt roskia, esim. kuvioiden esimerkissä solu 8 on muuttunut roskaksi, mutta nämä roskat kerätään seuraavilla keräyskerroilla.

- Edellä olevasta havaitaan, että välittömästi keräyksen jälkeen kypsä
- 5 sukupolvi on täsmälleen sama keskusmuistissa ja levyllä, mutta sitä mukaa, kun juuri kerättyä sukupolvea vanhempia sukupolia kerätään, päivitetään nuoremmissa sukupolvissa olevat viittaukset osoittamaan kohti vanhempien sukupolvien solujen uutta paikkaa. Tämä viittausten uudelleen ohjaus tapahtuu keksinnön mukaisesti vain keskusmuistissa, ei levyllä. Levyllä olevassa tieto-
- 10 kantakuvaassa ovat nuoremasta vanhempaan sukupolveen olevat osoittimet siis vanhentuneita, koska ne osoittavat levyllä sellaisiin paikkoihin, jotka on keskusmuistin puolella jo vapautettu. Kun esim. sukupolvi D' oli kirjoitettu levylle, käyttiin D':ssa olevaa dataa (osoitinarioja) vielä muuttamassa keskusmuistissa, mutta ei levyllä. Levyllä olevien tietojen (osoittimien) päivitystä ei
- 15 haluta tehdä seuraavan (vanhemman) sukupolven keräyksen yhteydessä, koska juuri tällaiset päivitykset aiheuttavat ylimääräistä levyliikennettä ja ovat erityisen hitaita, koska ne vaativat pieniä päivityksiä eri kohtiin (kirjoituspäään toistuvaa siirtelyä). Näin ollen on levyllä olevasta tietokantakuusta pystytävä päättelämään se tilanne, joka keskusmuistissa on ollut vikaantumishetkellä.
- 20 Esillä olevan keksinnön mukainen toipumismekanismi pystyy rekonstruoimaan ajonaikaiset vanhemmat sukupolvet suorittamalla keräbyn sukupolven viimeisintä versiota edeltävälle levykopiolle roskankeruun kaltaista toimintaa, jonka avulla vanhentuneet osoittimet saadaan korjattua oikeiksi. Toipumisen aikana suoritetaan jokaiselle kerätylle sukupolvelle tällainen operaatio alkaen nuorimmasta sukupolvesta kohti vanhempa, jolloin koko tietokannan tila saadaan lopulta identtiseksi siihen nähdien mitä se oli juuri ennen vikatilannetta. Muistojoukkoja ei kirjoiteta lainkaan levylle, vaan toipumisprosessi osaa päättellä ne itse. Keskusmuistissakin muistojoukot ovat elossa vain hetken aikaa.
- 25
- 30 Kuviossa 7 on havainnollistettu esimerkkiä edellä kuvatusta ongelasta, joka on ratkaistava toipumisprosessissa. Kuvio vastaa muuten kuvion 6e tilannetta, mutta esimerkin havainnollisuuden vuoksi oletetaan, että solut 7...9 ovat sukupolvessa E (levyllä) niin, että solu 8 osoittaa soluun 5 ja solu 9 soluun 6. Esimerkissä oletetaan siis, että E on nuorin kypsä sukupolvi. Edellä kuvatun keruun seurauksena tilanne keskusmuistissa oli ennen vikaantumista sellainen kuin kuviossa on esitetty yhtenäisillä viivoilla ja levyllä puolestaan
- 35

katkoksen jälkeen sellainen kuin on esitetty katkoviivoilla. Sähkökatkoksen jälkeen keskusmuisti on tyhjä, jolloin levylä olevien tietojen perusteella keskusmuistiin on palautettava em. yhtenäisillä viivoilla esitetty tietokantakuva. Toisin sanoen, levylä olevassa tietokantakuvassa ovat nuoremmasta vanhempaan sukupolveen osoittavat osoittimet vanhentuneita, sillä ne osoittavat sukupolviin, joiden keräystä ei ole suoritettu. Nämä osoittimet tulisi siis päivittää oikeiksi toipumisen yhteydessä. (Huomattakoon, että koska D oli nuorin kerätty sukupolvi, keräys teki vain D:hen ja sitä vanhempaan sukupolviin osoittavista osoittimista vanhentuneita (ei-valideja) ja esim. sukupolvista F ja G lähtevät osoittimet ovat voimassa, koska ne on talletettu kypsäksi sukupolveksi heti keräyksen jälkeen, eivätkä niiden osoittimet ole sen jälkeen muuttunut keskusmuistissakaan.)

Toipuminen alkaa siitä, että tyhjään keskusmuistiin luetaan levyltä sukupolvet A, B, C ja D (eli ko. sukupolia vastaavat sivut) samoihin paikkoihin, joissa ne olivat ennen vikaa. (Levylä on metadataa, joka kertoo, missä ko. sukupolvien sijainnit.) Kuten edellä esitettiin, sukupolia A...D ei vielä vapautettu levyltä.

Nämä sukupolvet merkitään sellaisiksi, että ne tullaan keräämään (keräys tapahtuu toipumisen yhteydessä ja sen tarkoituksesta on rekonstruoida vanhentuneet osoittimet oikeiksi). Huomattakoon tässä yhteydessä, että levylä olevissa sukupolvissa A...D on vielä ne roskat jäljellä, jotka edellä kerättiin sukupolvista A'...D'. Toipuminen koostuu osaksi datan lukemisesta levyltä ja osaksi roskankeruun kaltaisesta toiminnasta, jonka yhteydessä rekonstruoidaan muistojoukot sellaisiksi kuin ne olivat keskusmuistissa juuri ennen vikatiannetta. Kun muistojoukot on saatu oikeaan muotoon, pystytään vanhentuneet osoittimet muuttamaan niin, että ne osoittavat oikeisiin paikkoihin.

Kun sukupolvet A...D on luettu levyltä ja merkitty kerättäviksi, luetaan vielä sukupolvi E keskusmuistiin, jolloin keskusmuistissa ollaan kuvion 8a esittämässä tilanteessa. Tämän jälkeen käydään kaikki sukupolven E solut läpi sen toteamiseksi, onko E:stä osoittimia sellaisiin sukupolviin, jotka on merkitty kerättäviksi. Kun tällaisia soluja havaitaan, rekonstruoidaan kyseisen sukupolven muistojoukkoa.

Tässä esimerkkitapauksessa havaitaan näin ollen aluksi, että solusta 7 on osoitin soluun 6, jolloin sukupolven D muistojoukkoon RS\_D lisätään ensimmäiseksi osoite siihen kohtaan solua 7, josta on osoitin soluun 6. Seuraavaksi D:n muistojoukkoon lisätään samalla tavalla osoitin siihen koh-

taan solua 8, josta on osoitin soluun 5 ja viimeiseksi D:n muistojoukkoon lisätään edelleen samalla tavalla osoitin siihen kohtaan solua 9, josta on osoitin soluun 6. Sukupolven D muistojoukko on nyt saatu rekonstruoitua sellaiseksi kuin se oli ennen vikatilannetta. Näin ollaan kuvion 8b mukaisessa tilanteessa.

Tässä vaiheessa luetaan sukupolvi D' levyltä samaan paikkaan kuin missä se oli aikaisemmin ja käydään läpi D' solu solulta forward-osoitimien muodostamiseksi sukupolvesta D sukupolveen D'. Läpikäynti suoriteaan käymällä läpi D:n muistojoukkoa samassa järjestyksessä kuin roskantereun yhteydessä ennen vikatilannetta. D':n läpikäytiä varten asetetaan osoitin SP sukupolven ensimmäisen solun alkuun (kuva 8c). Osoitinta siirretään tämän jälkeen eteenpäin seuraavassa kuvattavalla tavalla. Kun käydään muistojoukon RS\_D ensimmäinen alkio läpi, huomataan, että se osoittaa solussa 7 olevaan osoittimeen, joka osoittaa soluun 6 sukupolvessa D. Sukupolvessa D ei ole forward-osottimia, koska se luettiin juuri levyltä, joten nyt muodostetaan forward-osoitin sukupolven D' ensimmäiseen soluun. (Solujen järjestys tiedetään aikaisemman roskankeron perusteella.) Samalla muutetaan solusta 7 löydetty osoitin osoittamaan soluun 6'. Tämän jälkeen viedään osoitinta PT eteenpäin seuraavan solun (5') alkuun (merkitty katkoviivalla kuviossa 8c) ja luetaan muistojoukosta RS\_D seuraava alkio. Tämä osoittaa soluun 8, josta on osoitin soluun 5, jolloin muodostetaan forward-osoitin solusta 5 soluun 5'. Samalla muutetaan solusta 8 löydetty osoitin osoittamaan soluun 5'. Tämän jälkeen viedään osoitin PT jälleen eteenpäin yhden solun verran ja luetaan muistojoukosta seuraava alkio. Tämä osoittaa soluun 9, josta on osoitin soluun 6. Nyt havaitaan, että solussa 6 on jo forward-osoitin, jolloin yksinkertaisesti sijoitetaan solun 9 osoittimen arvoksi solun 6 forward-osoitimen arvo. Silloin, kun solusta löydetään forward-osoitin, osoitinta PT ei viedä eteenpäin.

Näin on saatu viimeinenkin sukupolven E sukupolveen D osoittavista osoittimista rekonstruoitua vikaa edeltävään tilaan. Tällä tavoin muutetaan jokainen nuoremman sukupolven (E) osoitin osoittamaan oikeaan vanhemman sukupolven (D') soluun. Päivitys tapahtui rekonstruoimalla ensin nuorimman "kerättävän" sukupolven (D) muistojoukko nuoremman sukupolven avulla ja käymällä tämän jälkeen läpi muistojoukkoa samaan tapaan kuin varsinaisen roskankeräyksen yhteydessä. Sukupolvi D ja sitä vastaavat sivut voidaan tämän jälkeen vapauttaa.

Tämän jälkeen kootaan seuraavaksi samalla tavalla sukupolven C muistojoukko eli käymällä läpi kerätyn sukupolven (D') kaikki solut samalla tavalla kuin edellä tehtiin sukupolvelle E ja päivittämällä vanhempien keräämättömiensukupolvienmuistojoukkoja. Kun sukupolven C muistojoukko on 5 saatu muodostettua, luetaan C' levyltä ja muodostetaan forward-osoittimet sukupolvesta C sukupolveen C' sekä muutetaan D':n sukupolveen C osoittavat osoittimet osoittamaan sukupolveen C'. Tämän jälkeen käydään taas läpi kerätyn sukupolven (C') kaikki solut ja päivitetään vanhempien keräämättömiensukupolvienmuistojoukkoja, jne. Kun vastaavat operaatiot on suoritettu 10 sukupolville B ja A, on oikea tietokantakuva saatu rekonstruoitua.

Edellä kuvatulla tavalla saadaan ajonaikaiset sukupolvet rekonstruoitua "vertailemalla" kerätyn sukupolven viimeisintä ja sitä edeltävää levykopioita keskenään.

Edellä kuvatuissa operaatioissa oletetaan, että järjestelmällä on 15 joitakin perusominaisuksia, jotta edellä kuvatut toiminnallisuudet saadaan toteutettua. Ensinnäkin, osoittimet on voitava tunnistaa soluista eli solujen muun sisällön joukosta on pystyttävä löytämään osoittimet (ja tunnistettava forward-osoittimet). Toisaalta on myöskin voitava tunnistaa solurajat eli kunkin solun pituus.

20 Toipumisprosessia varten sukupolia A...D ei välttämättä tarvitse lukea levyltä keskusmuistiin, jos kaikki osoittimet niitä nuoremmissa sukupolvissa voidaan luetella katsomatta muihin soluihin eli jos solut kuvaavat itsensä.

Koska muistojoukkoihin tehdään lisäksiä kypsien sukupolvien ke-ruun yhteydessä silloin, kun solua kopioitaessa havaitaan, että siitä on osoitin 25 soluun, joka on vielä keräämättömässä sukupolvessa, ja koska toivuttaessa käydään läpi uusi sukupolvi ja päivitetään vanhempien keräämättömiensukupolvienmuistojoukkoja, ovat muistojoukkojen sisällöt kussakin vaiheessa aina samat ennen toipumista ja toivuttaessa. Muistojoukkojen päivitystä ei kuiten-kaan tarvitse välttämättä tehdä heti kopioitaessa, vaan se voidaan tehdä 30 erillisessä läpikäynnissä jälkeenpäin, esim. toivuttaessa. Edellä esitetty tapa on kuitenkin nopeampi.

Oleellista edellä kuvattussa menetelmässä on, että käytäessä läpi muistojoukkoja niissä olevien osoittimien osoittamat solut kopioidaan deter-ministisiin paikkoihin niin, että solut ovat toipumisen jälkeen samoilla paikoilla 35 kuin ennen vikaantumista. Periaatteessa on esim. mahdollista aloittaa solujen kopointi uuteen sukupolven sen lopusta päin, jos muistissa on tieto sukupol-

ven koosta. Vastaavasti voitaisiin uusi sukupolvi käydä läpi lopusta alkuun päin vanhempien keräämättömiensukupolvien muistojoukkojen päivittämiseksi. Muistojoukkojen rekonstruointi voi siis tapahtua useilla tavoilla, kunhan se tapahtuu toipumisen yhteydessä ja ennen sitä toisiaan vastaavilla tavoilla.

- 5 Muistojoukon osoittamien osoittimien muuttaminen osoittamaan mainitun viimeisimmän version soluihin voitaisiin toteuttaa myös esim. siten, että forward-osoittimet talletetaan erilliselle muistialueelle, joka on sukupolven edellisen version (esim. D) mittainen ja jossa forward-osoitin on vastaavassa kohdassa kuin missä se olisi ko. sukupolvessa (D). Tämän ratkaisun etuna on  
10 se, että sukupolvi pysyy samanlaisena koko roskankeroon ajan, jolloin sitä voidaan käyttää järjestelmissä, joissa useampi kuin yksi prosessori suorittaa roskankeroon rinnakkain.

- Vaikka keksintöä on edellä selostettu viitaten oheisten piirustusten mukaisiin esimerkkeihin, on selvää, ettei keksintö ole rajoittunut siihen, vaan  
15 sitä voidaan muunnella oheissakaan patenttivaatimuksissa esitetyn keksinnöllisen ajatuksen puitteissa. Menetelmää voidaan esimerkiksi käyttää monissa erilaisissa sovellusympäristöissä. Kuten edellä todettiin, suurin hyöty menetelmästä saavutetaan sellaisessa varmentavassa ympäristössä, jossa reaalialaisuus on tärkeää. Em. sovellusalueiden lisäksi keksintöä voidaan hyödyntää esim.  
20 teollisuuden prosessinvälinnassa.

### Patenttivaatimukset

1. Menetelmä levyvarmennetun tietokannan elvyttämiseksi, jossa menetelmässä
  - ylläpidetään keskusmuistissa tietokantaa, joka käsittää ensimmäisen sukupolven ja ainakin yhden kypsän sukupolven, jotka sukupolvet sisältävät muistisoluja, joihin on talletettu dataa ja lisäksi osoittimia, jotka muodostavat muistisolujen välisiä viitauksia,
- 5      - ylläpidetään kypsien sukupolvien alueella keskusmuistissa sukupolviointaisia muistojoukkoja, joissa luetellaan sukupolviointiesti niiden osoittimien osoitteet, jotka osoittavat kyseessä olevaan sukupolveen,
- 10     - alllokoidaan muistia sovellusohjelman käyttöön keskusmuistin ensimmäisen sukupolven alueelta,
- 15     - kerätään ensimmäisen sukupolven alueella elossa olevat muistisolut ajoittain uudeksi kypsäksi sukupolveksi keskusmuistiin kypsien sukupolvien alueelle,
  - suoritetaan kypsien sukupolvien alueella sukupolvittain roskankeuruuta, jolloin keruun yhteydessä käydään läpi muistojoukkoja ja kopioidaan elossa olevia muistisoluja muistojoukon osoittamassa järjestyksessä ajallisesti uudempaan kypsään sukupolveen,
- 20     - tallennetaan kypsä sukupolvi roskankeerun jälkeen erilliseen levymuistiin,
  - roskankeerun edetessä kypsien sukupolvien alueella tehdään muutoksia sukupolvien välisiin viitauksiin sellaisessa sukupolvessa, joka on jo talletettu levymuistiin,
- 25     t u n n e t t u siitä, että
  - ainakin osa mainituista muutoksista tehdään vain keskusmuistissa, levymuistissa ylläpidetään kypsästä sukupolvesta viimeisimmän levylle talletetun version (D') lisäksi sitä edellistä levylle talletettua versiota (D), ja elvyttäminen suoritetaan mainittujen versioiden avulla (a) rekonstruoimalla mainitun edellisen levylle talletetun version muistojoukko, joka osoittaa ne osoittimet, joista on viitauksia kyseiseen sukupolveen ja (b) muuttamalla muistojoukon osoittamat osoittimet osoittamaan mainitun viimeisimmän version muistisoluihin.
- 35     2. Patenttivaatimuksen 1 mukainen menetelmä, t u n n e t t u siitä, että elvyttämisen yhteydessä

asetetaan apuosoitin (PT) osoittamaan mainitun viimeisimmän version (D') ensimmäiseen muistisoluun,

- 5 käydään läpi muistojoukko samassa järjestyksessä kuin roskaneruu yhteydessä, jolloin (i) kun muistojoukon läpikäynnin yhteydessä tullaan ensimmäistä kertaa tiettyyn muistisoluun mainitussa edellisessä versiossa (D), muodostetaan osoitin osoittamaan kyseistä muistisolu vastaavasta muistipaikasta apuosoittimen osoittamaan muistisoluun, muutetaan muistojoukon osoittama osoitin osoittamaan apuosoittimen osoittamaan muistisoluun ja siirretään apuosoitinta muistisolun verran eteenpäin mainitussa viimeisimmäsä versiossa, ja (ii) kun muistojoukon läpikäynnin yhteydessä tullaan uudelleen tiettyyn muistisoluun, annetaan muistojoukon osoittamalle osoittimelle kyseistä muistisolu vastaavassa muistipaikassa jo olevan osoittimen arvo ja jätetään siirtämättä apuosoitinta, ja
- 10 käydään läpi kerätyn sukupolven (D') kaikki muistisolut ja päivitetään vanhempien keräämättömien sukupolvien muistojoukkoja.

- 15 3. Patenttivaatimuksen 2 mukainen menetelmä, t u n n e t t u siitä, että kun muistojoukon läpikäynnin yhteydessä tullaan ensimmäistä kertaa tiettyyn muistisoluun mainitussa edellisessä versiossa (D), kirjoitetaan mainittu osoitin suoraan kyseiseen muistisoluun, jolloin se osoittaa kyseisestä muistisolusta apuosoittimen osoittamaan muistisoluun, jolloin, kun muistojoukon läpikäynnin yhteydessä tullaan uudelleen tiettyyn muistisoluun, annetaan muistojoukon osoittamalle osoittimelle kyseisessä muistisolussa jo olevan osoittimen arvo.

- 20 4. Patenttivaatimuksen 1 mukainen menetelmä, t u n n e t t u siitä, että muistojoukkoja talletetaan keskusmuistiin vain keräämättömien sukupolvien yhteyteen.

- 25 5. Patenttivaatimuksen 4 mukainen menetelmä, t u n n e t t u siitä, että kun minkä tahansa sukupolven keruun yhteydessä havaitaan, että kerätyssä muistisolussa on osoitin muistisoluun, joka on sukupolvessa, joka on merkitty kerättäväksi, lisätään kyseisen sukupolven muistojoukkoon kyseisen osoittimen osoite.

- 30 6. Patenttivaatimuksen 1 mukainen menetelmä, t u n n e t t u siitä, että roskankeruu kypsien sukupolvien alueella suoritetaan nuorimmista sukupolvista kohti vanhempia sukupolia.

- 35 7. Patenttivaatimuksen 6 mukainen menetelmä, t u n n e t t u siitä, että kypsän sukupolven keruussa

allokoidaan sukupolvesta uusi versio kypsien sukupolvien alueelle, käydään läpi kerättävän sukupolven muistojoukko siten, että muistojoukon osoittamien osoittimien osoittamat muistisolut kopioidaan muistojoukon läpikäytijärjestyksessä mainittuun uuteen versioon, ja

5           käydään läpi muistin juurilohko siten, että ne muistisolut, joihin juurilohkosta on osoittimet kopioidaan mainittuun uuteen versioon.

8. Patenttivaatimuksen 7 mukainen menetelmä, t u n n e t t u   siitä, että kun kypsän sukupolven keruun yhteydessä havaitaan, että kopioidussa muistisolussa on osoitin muistisoluun, joka on sukupolvessa, joka on merkity 10       kerättäväksi, lisätään kyseisen sukupolven muistojoukkoon kyseisen osoittimen osoite.

9. Patenttivaatimuksen 8 mukainen menetelmä, t u n n e t t u   siitä, että roskankeroon aikana yhdistetään kaksi sukupolvea yhdeksi sukupolveksi, jos niiden yhteenlaskettu koko on pienempi kuin ennalta määritetty raja.

15

**(57) Tiivistelmä**

Keksintö koskee menetelmää levyvarmennetun tietokannan elvyttämiseksi. Keskusmuistissa ylläpidetään tietokantaa, joka käsitteää ensimmäisen sukupolven ja ainakin yhden kypsän sukupolven. Sukupolvet sisältävät muistisoluja, joihin on talletettu dataa ja lisäksi osoittimia, jotka muodostavat muistisolujen välisiä viitauksia. Kypsien sukupolvien alueella keskusmuistissa ylläpidetään sukupolvikohtaisia muistojoukkoja. Ensimmäisen sukupolven alueella elossa olevat muistisolut kerätään ajoittain uudeksi kypsäksi sukupolveksi keskusmuistiin kypsien sukupolvien alueelle. Kypsien sukupolvien alueella suoritetaan sukupolvittain roskankerousta, jossa kopioidaan elossa olevia muistisolua muistojoukon osoittamassa järjestyksessä ajallisesti uudempaan kypsään sukupolveen. Roskankeroon edetessä kypsien sukupolvien alueella tehdään muutoksia sukupolvien välisiin viitauksiin sellaisessa sukupolvessa, joka on jo talletettu levymuistiin. Järjestelmän tehokkuuden parantamiseksi ainakin osa mainituista muutoksista tehdään vain keskusmuistissa, levymuistissa ylläpidetään kypsästä sukupolvesta viimeisimmän levylle talletetun version lisäksi sitä edellistä levylle talletettua versiota, ja elvyttäminen suoritetaan mainittujen versioiden avulla (a) rekonstruoinalla mainitun edellisen levylle talletetun version muistojoukko ja (b) muuttamalla muistojoukon osoittamat osoittimet osoittamaan mainitun viimeisimmän version muistisoluihin.

(kuvio 3)

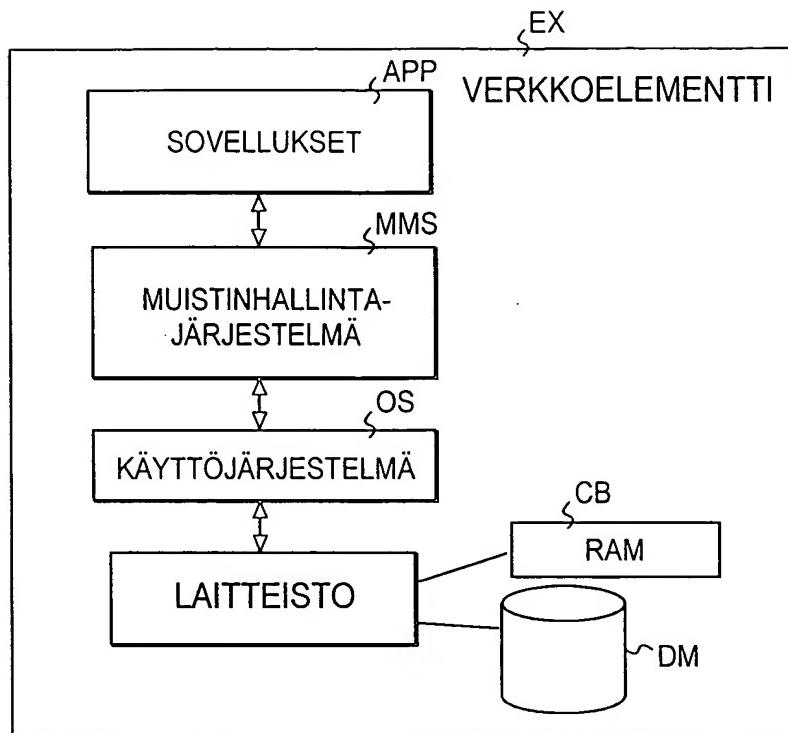
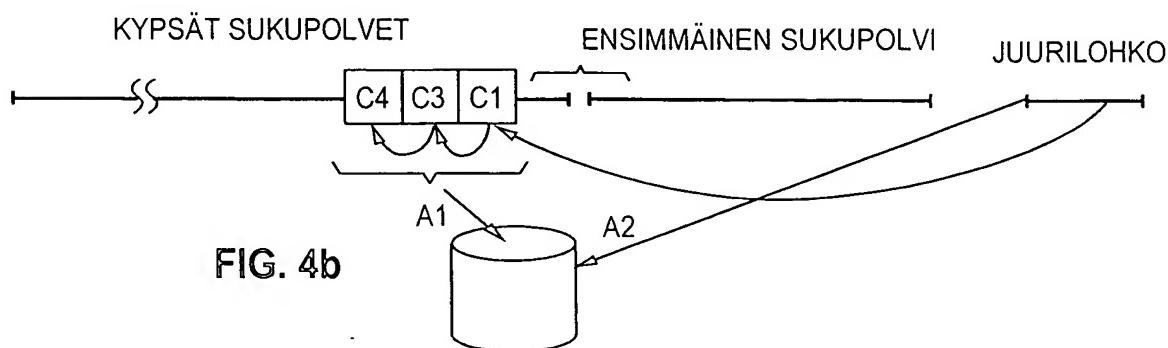
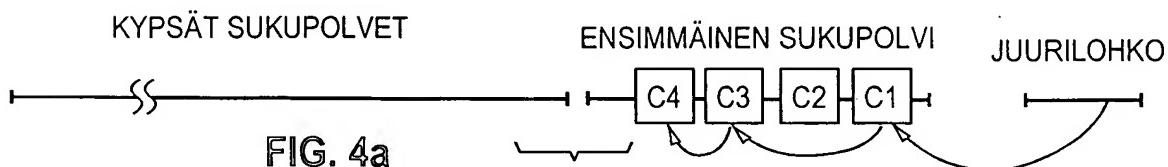
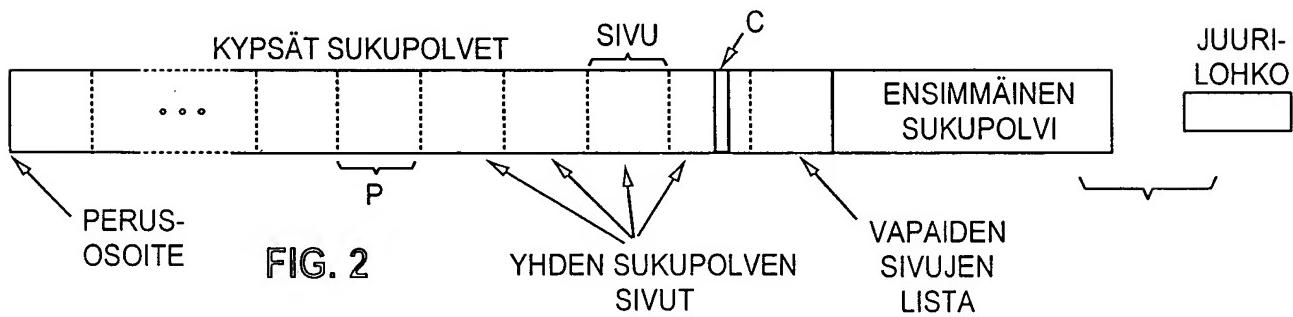
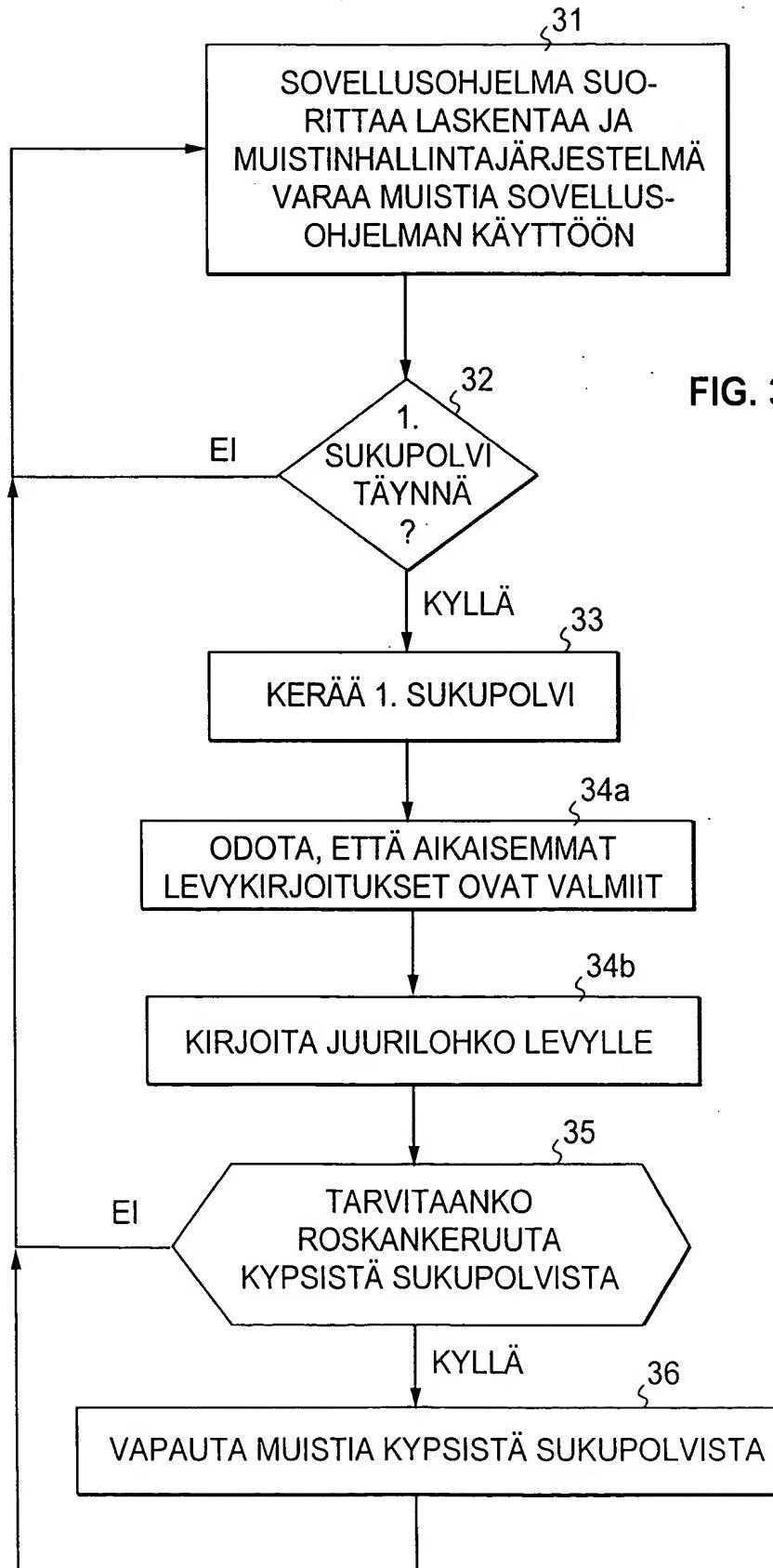
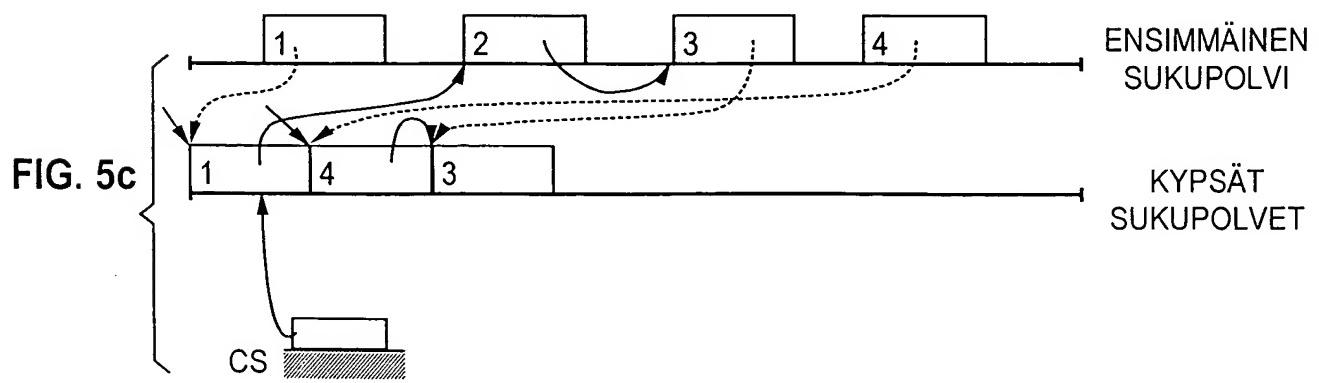
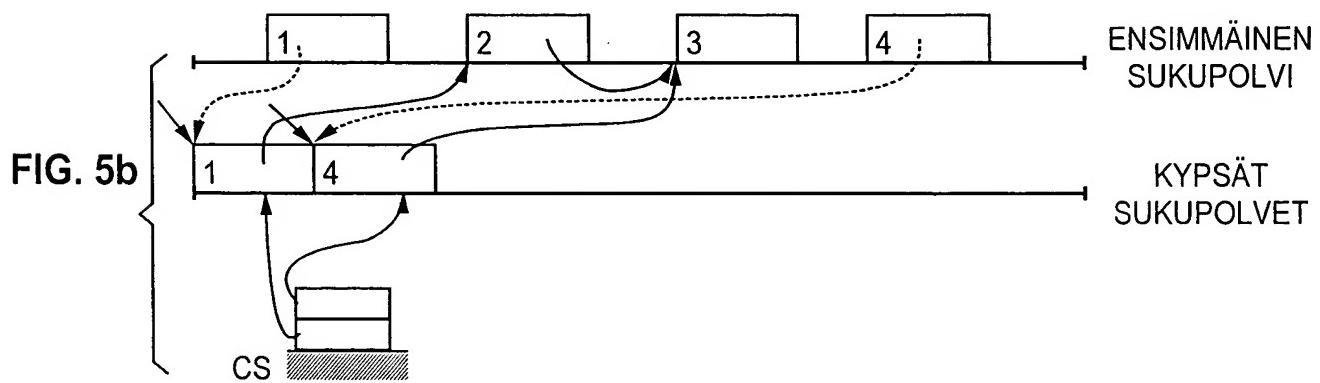
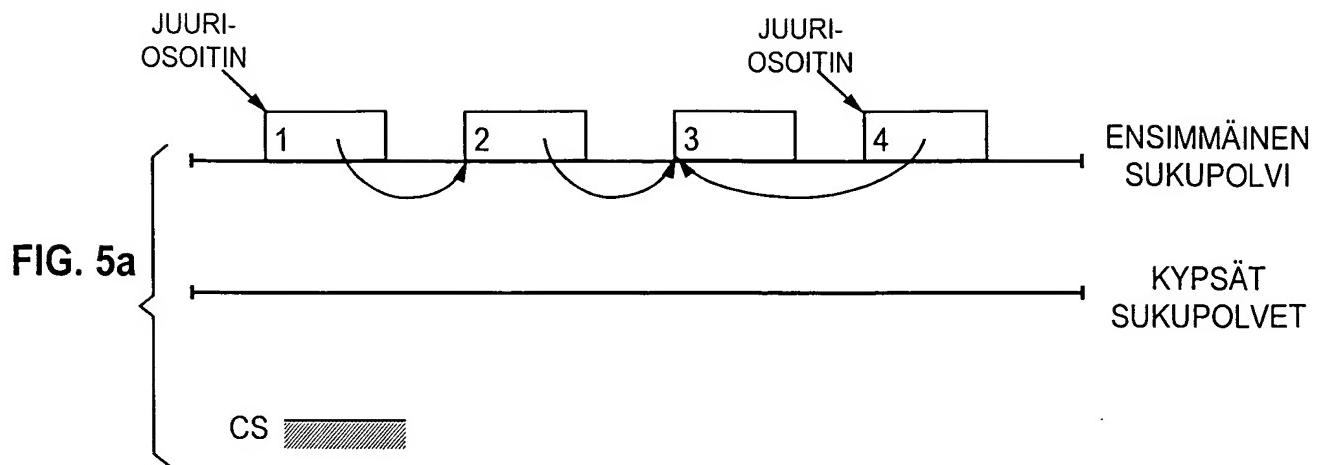
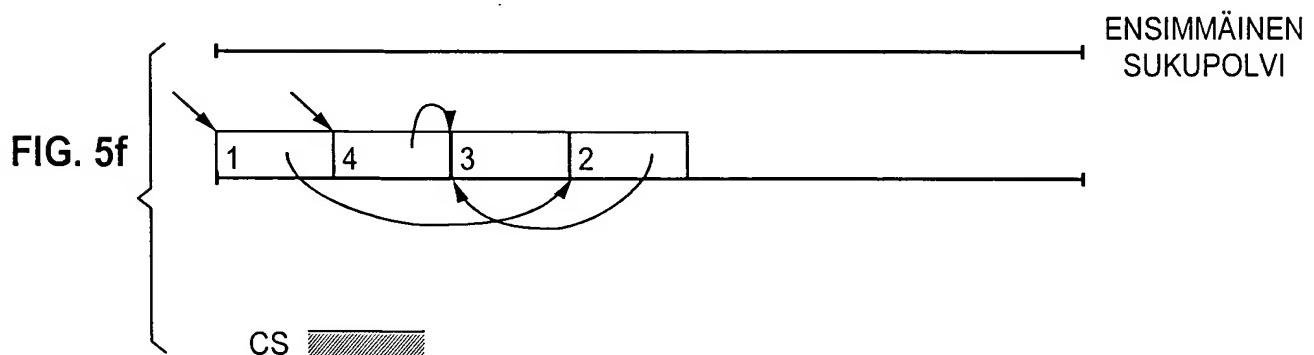
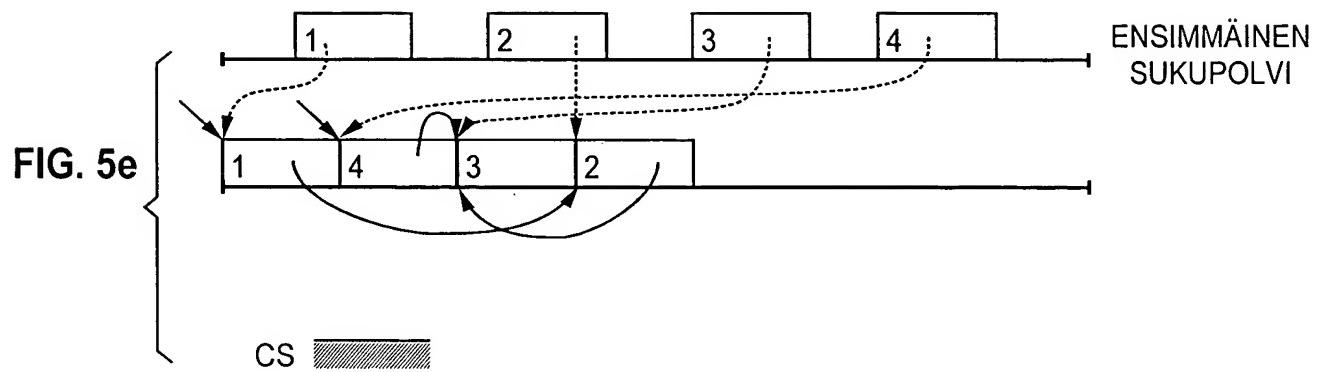
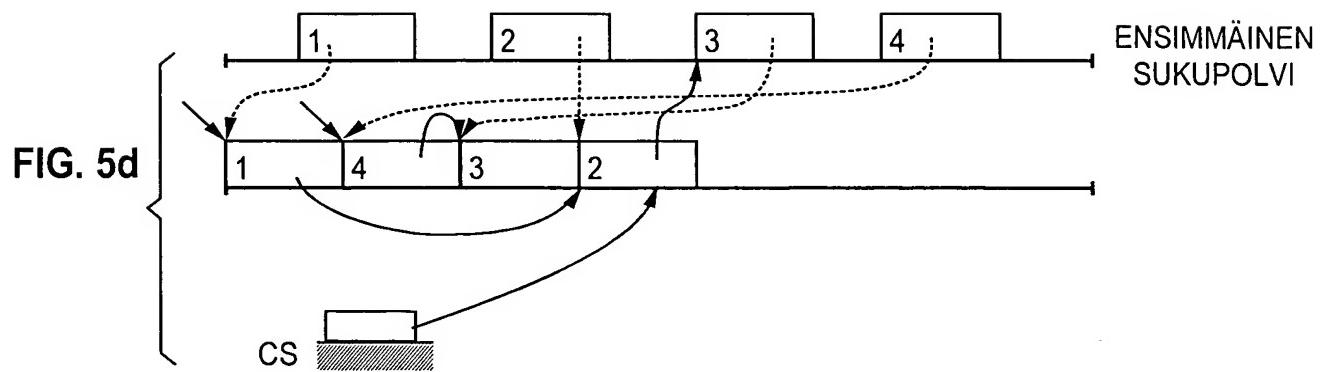


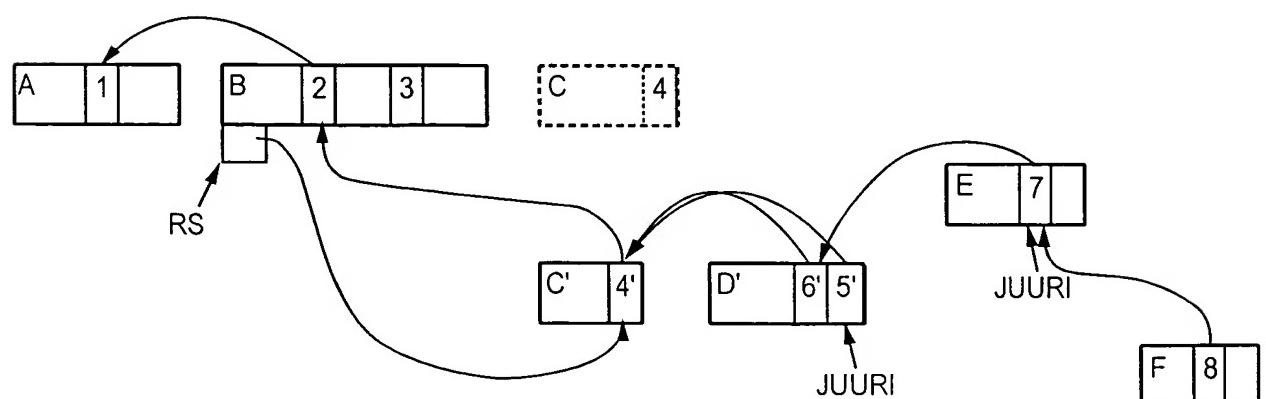
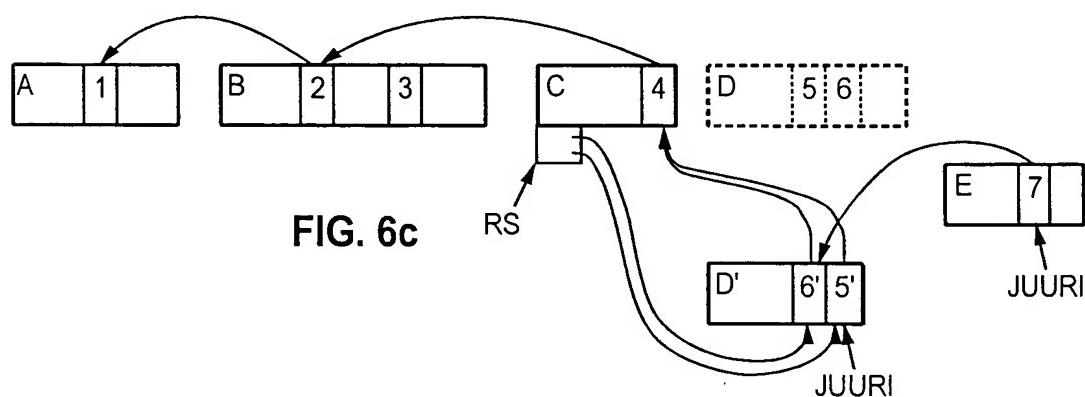
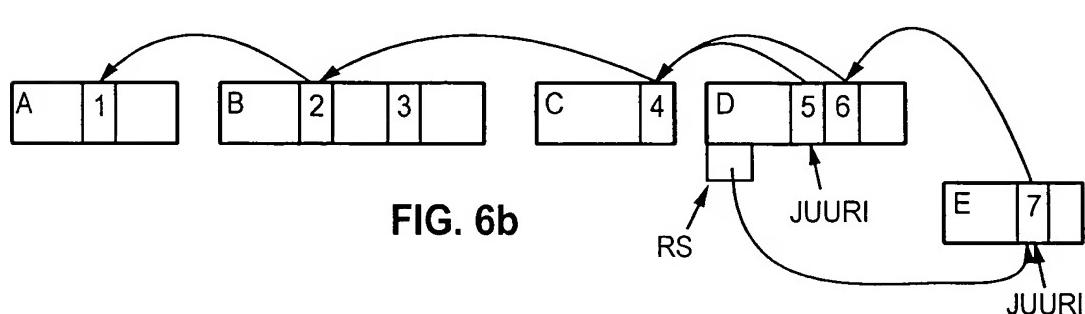
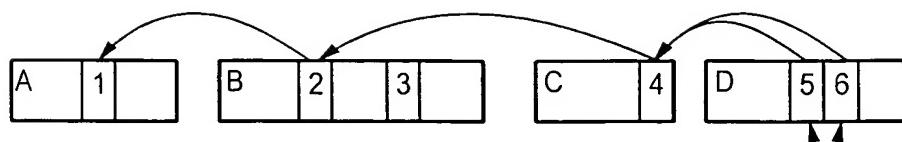
FIG. 1











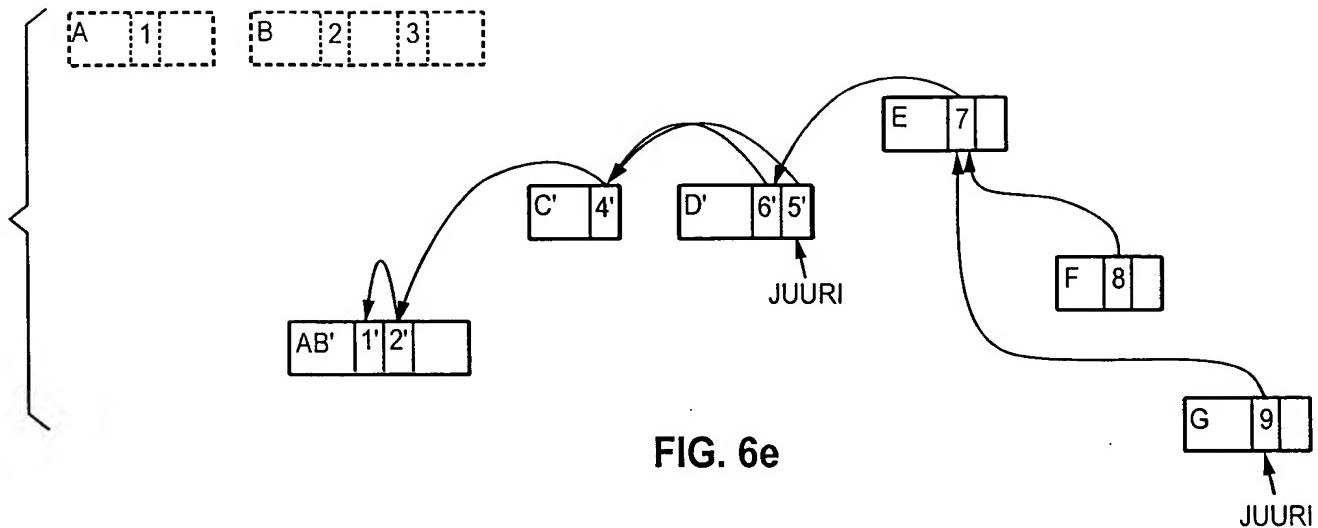


FIG. 6e

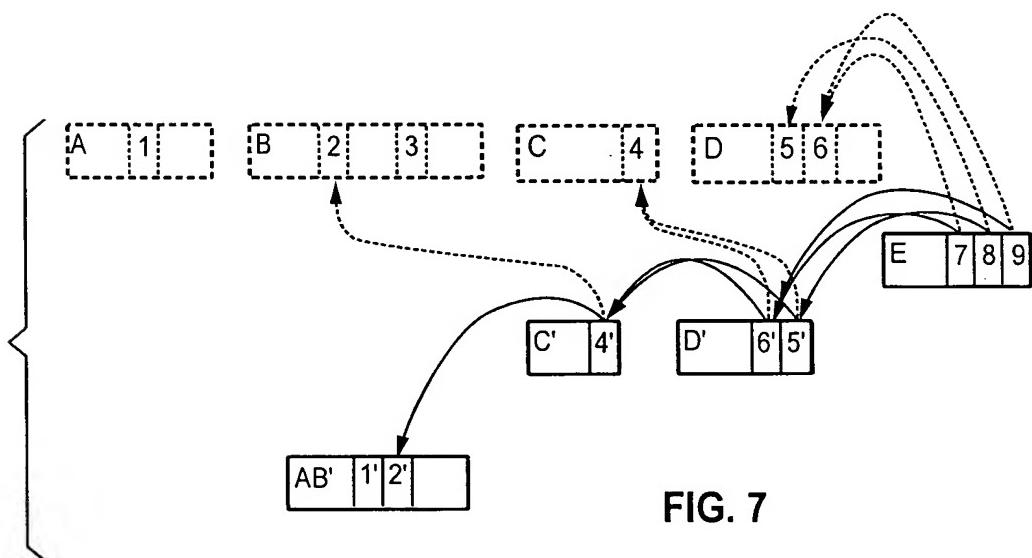


FIG. 7

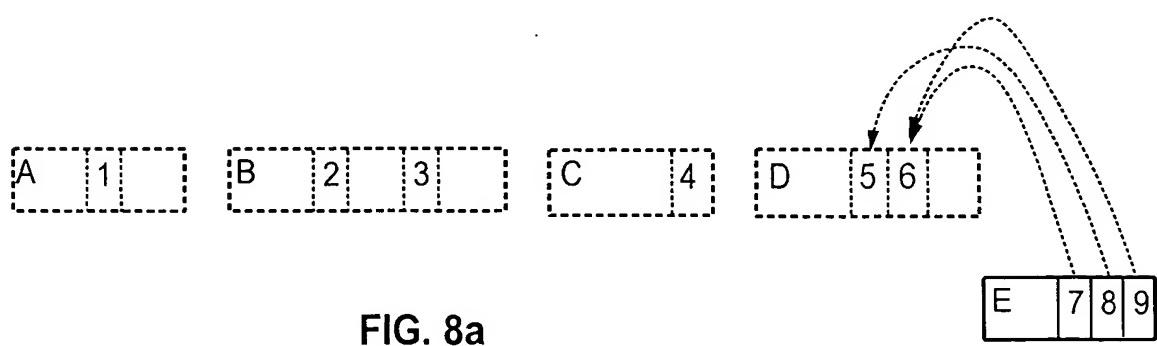


FIG. 8a

